THE INPUT OUTPUT UNIT FOR THE ATTITUDE AND ARTICULATION CONTROL SUBSYSTEM ON THE CASSINI SPACECRAFT

E. Shatom
Avionic Systems and Technology Division
Jet Propulsion Laboratory, California Institute of Technology
4800 **Oak Grove Drive**
Pasadena, CA 91109

ABSTRACT

The Input Output Unit (IOU) for the Attitude and Articulation Subsystem (AACS) for the Cassini spacecraft uses an embedded microprocessor to format and interpret data packets sent over a bus with the electrical characteristics of MIL-STD-1553B. The IOU used available design and protocol elements when possible, and employed custom hardware and firmware elements when necessary. As a result, the hardware design of the IOU, including the design of a custom gate array, took place in a extremely short time. With extensive simulation and modeling of the design at both the chip and board level, design iterations were minimal, and there were no iterations of the gate array.

The embedded microprocessor in the IOU provides great versatility and flexibility, and allowed the incorporation in many functions in firmware. For this reason, firmware design and verification were challenging linchpins of this effort.

This I/O approach is a marked departure from approaches used on previous JPL spacecraft. It has resulted in significant changes in the interfaces of AACS peripherals and their integration at the subsystem level. Future trends are reinforcing this approach, with "smart" peripherals and instruments communicating over much higher bandwidth optical buses.

INTRODUCTION

The Cassini spacecraft, being developed by the Jet Propulsion Laboratory (JPL) for the National Aeronautics and Space Administration (NASA), is planned for launch in October 1997. Cassini will deliver a probe into Saturn's largest moon, Titan, **and** spend four years touring the planet and its moons.

AACS Flight Computer (AFC)

In flight, the operation of the Cassini Attitude and Articulation Control Subsystem (AACS) is directed by AACS flight software executing in the Engineering Flight Computer (EFC). The EFC is a centralized, radiation-hard, 1750-based flight processor that has been procured from IBM. The EFC has been enhanced with additional functions to form the AACS Flight Computer (AFC). AFC functions include communicating with various AACS sensors and actuators (or "peripheraJs") dispersed over the spacecraft, communicating with the Command and Data Subsystem (CDS), and inputting serial imaging data from the AACS Stellar Reference Unit (SRU) imaging tracker.

The Input-Output Unit (IOU)

The key building block for implementing the communication between the AFC and AACS peripherals is the Input Output Unit (IOU), which is supplied to each peripheral. For the first time on a JPL spacecraft, this AACS communication employs digital data packets, or "messages". These messages, that support a broad range of functions, are interpreted by a microcomputer embedded in the IOU **executing a common firmware** program, An adapted IOU, designated as the Bus Controller IOU (BC IOU), is contained within the AFC and functions as an 1/0 processor for communication with the multiple peripheral IOUS (The BC IOU uses the same gate array as the IOU, but has custom firmware).

DESIGN APPROACH

The design of 1/0 for AACS peripherals on previous spacecraft has often been a daunting task. For example, on the Galileo spacecraft, launched to

Jupiter in 1987, a combination of custom parallel and seriat interfaces were used, as illustrated in Fig. 1, "Galileo AACS I/0". The parallel interfaces required bulky and massive cabling, while the serial interfaces utilized several different clock rates and protocols. The AACS flight software required unique interface drivers for each peripheral to reflect differences at the physical layer. The lack of visibility into I/O transactions and communication errors compounded the difficult task of providing fault protection functions for the spacecraft, Fig. 2, "Cassini AACS I/0", based upon the use of an I/O bus and IOUs, is presented for comparison.

Word and Message Protocol

A fundamental design decision for Cassini was the usage of the Manchester Encoding-Decoding word-level protocol in combination with a transformer isolated bus. Both of these elements, as used in the 1553B bus, provide excellent robustness, error detection capability, and fault isolation. However, it was also determined that utilizing the 1553B word-level protocol without some degree of "smart" electronics was inadequate for AACS in terms of visibility, additional functions, design flexibility, as well as in-flight reprogramming. In addition, at the time this decision was made, AACS I/O requirements included the transmission of large blocks of imaging pixel data that could not be accommodated in real time with the 32-word block limit constraint of the 1553B message protocol. These considerations led to the use of a "smart" IOU with a custom message protocol,

Firmware Functions

It was **determined that the best wa y to provide the "smart" electronics was with a microprocessor,** since it could support the generation and checking of data packets, as well as provide ancillary functions required by AACS. To some extent, the presence of the microprocessor spurred the generation of additional uses for it, so that ultimately it supported diverse features such as bus time-outs and switching, sending and interpreting messages, checking and executing commands (including delayed command execution), supporting a systematic interlocking "handshake" for data exchange with the host electronics (including "time-outs" if the host fails to respond), and even "throttling" the bus data rate in order to reduce power consumption.

Hardware Functions

In addition to requirements that could safely and effectively be embedded in IOU firmware, a number of functions were levied upon IOU hardware. As there was no need in this application to comply with the 1553B bit allocation for the command synch word, one of the bits in this word was chosen to be a "Clamp Bit", so that an IOU receiving a command synch word with this bit "set" would go into a continuous reset or "clamp" state (independent of the microprocessor), and in addition hold it's host electronics in the same state. Since a broadcast address for the AACS bus was also defined, this allowed the AFC to put all active peripherals into a clamp state by issuing a single word on the two bus channels. The AFC also has the option of forcing one (or all peripherals) onto either bus, as well as preventing the microprocessor from changing the IOU's bus channel. Another hardware feature is an "anti-babble" function that prevents an IOU from talking continuously (or ping-ponging) on both bus channels.

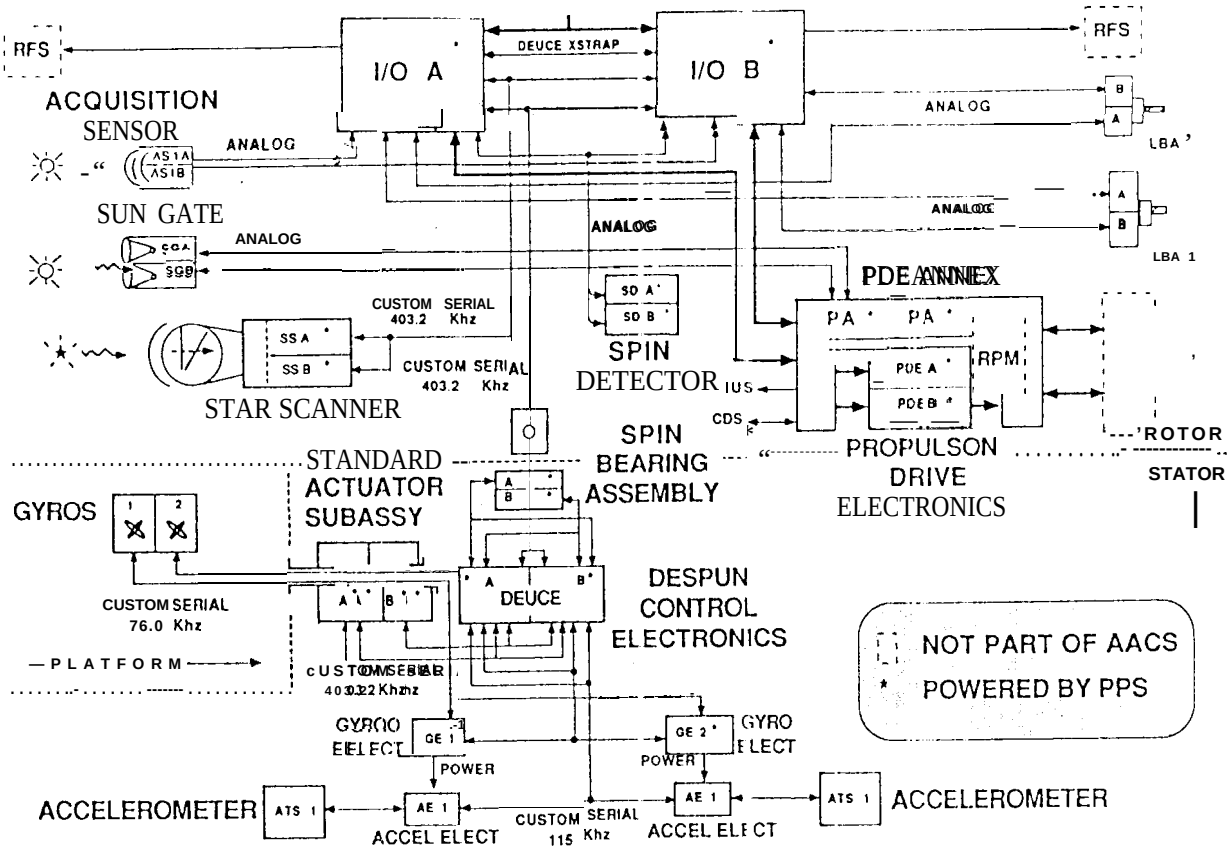
IMPLEMENTATION

Implementing all of the IOU requirements in discrete ICS requires about 100 additional components in addition to basic components such as the microprocessor, oscillator, RAM & PROM chips: in fact, two early breadboard units were built and tested with discrete parts. Flying 16 discrete IC units would have been a great challenge from the viewpoint of mass, volume, and power.

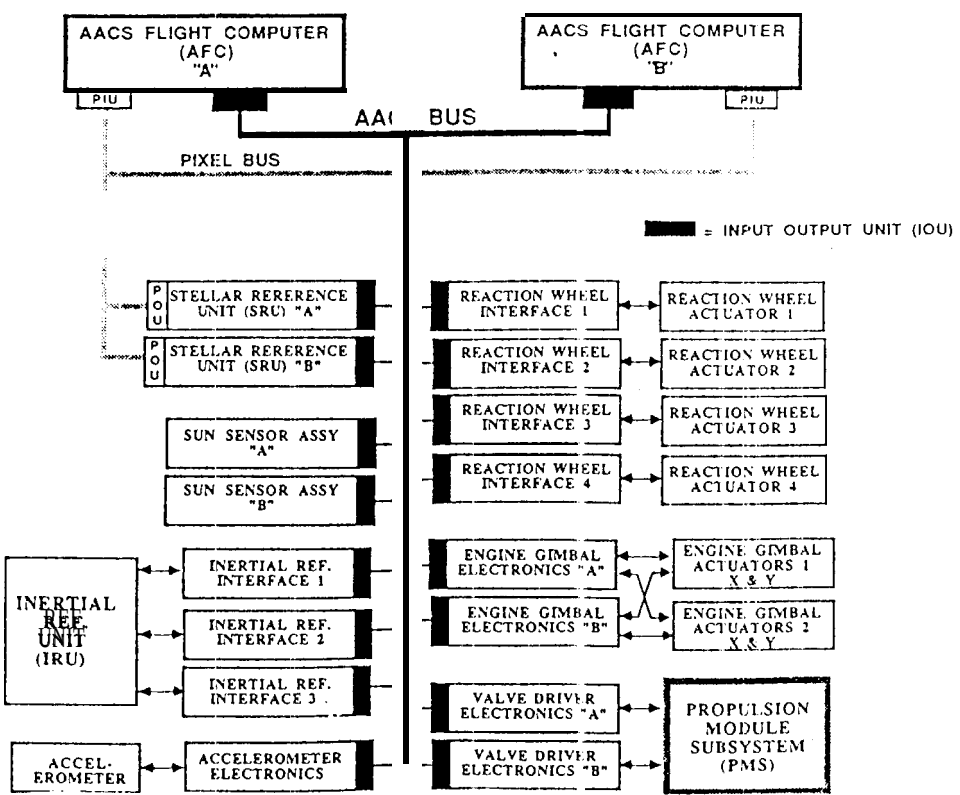
Gate Array Implementation

Fortunately, with the support and encouragement of the Cassini Spacecraft Office, radiation-hard gate arrays were approved for use in this application, allowing a significant reduction in volume, mass, and power. In particular, the availability of a Manchester-Encoding Decoding (MED) macro on a United Technologies Micro Electronics (u'fMC) gate array proved to be a tremendous benefit: not only was this macro available "off-the-shelf", but it had been used and qualified in other UTMC products, In addition, the designer of the this UTMC macro was available to help convert the discrete IOU design to a gate-level design (and interface the MED macro to the rest of the IOU), and also helped achieve 98.570 fault coverage for the gate array.

Figure 1: Galileo AACS I/O



. Figure 2: **Cassini AACS 1/0**



The entire gate array design time (from the start of conversion to the gate array PDR/CDR) was about 7 weeks. A block diagram of the IOU based upon the gate array is given in Fig. 3, "IOU Block Diagram".

Flight Design Implementation

After the validation of the gate array, extensive follow-on engineering **work was required in the areas of thermal design, layout and routing, and packaging design to achieve the final flight configuration shown in** Fig. 4, "Flight AACS IOU".

The flight IOU has the dimensions 7.0" x 2.95" x 0.74" and a mass of 0.23 Kg, It consumes about 0.5 watts in the standby mode (either quiescent or receiving) , and less than 1.0 watts when actively transmitting (for the AACS duty cycle, transmit power is about 0.6 watts).

FIRMWARE DEVELOPMENT

Since the operation of the IOU is so closely coupled to the firmware, which provides much of the functionality of the unit, much effort was required in identifying and documenting firmware requirements, writing, documenting and testing the code, and validating the code.

I/O Bus Bandwidth

Early on, a decision had been made to usc programmed I/0, so that the processor would be directly involved in moving bytes of data on and off the bus. Even though it was recognized that this would put a limit on the available bus bandwidth, at ., the time it was felt that the available bandwidth was more than adequate to meet the subsystem requirements. However, due to a number of factors not defined at the time, such as interaction between the flight software and the I/O bus, bus bandwidth on the I/O did eventually become a concern. As a result, **extra effort was required in the** firmware effort to measure and deliver this bandwidth.

Development Environment

The validation of the operation of multiple IOUS on a common bus was addressed by a number of stratagems. The use of in-circuit emulators was essential. It also proved valuable to develop a "bootstrap" ROM that contain the minimal code necessary to receive a test program, and then load, execute, and debug it out of RAM.

FIELD EXPERIENCE

Assern bly-Level

Having a fixed I/O interface at the "user interface" (IOU to user hardware) constrained users, but also allowed the sharing of generic interface designs. Since the I/O bus I/F was standard, PC- based "Bus Controller IOU" (BC IOU) simulators were produced and made available to users. Users had the responsibility of developing their own test code to communicate with their assemblies. Personnel who had helped develop I/O drivers for one peripheral were often able to efficient y get another assembly up and running. Problems did arise related to timing bctwccn the host computer and the BC IOU Simulator, and/or not accommodating delays in the I/F between the User I/F and thè assembly, that required small corrections.

Subsystem Level

In the AACS subsystem-level test bed, or Integrated Test laboratory (ITL), the frequent integration of different hardware assemblies has been relatively smooth using the IOUS and the AACS bus. Connection to the AACS bus is based upon a simple waveform check; the integrity of the data packets is confirmed during functional test. The capability also exists to use an AACS Bus Monitor, a piece of support equipment that can capture and store several minutes of AACS Bus traffic, for independent verification and analysis.

OBSERVATIONS & LESSONS LEARNED

The direction taken in development of the IOU was unique for a large JPL in-house space project. The gate array development was expedited by the usc of an off-the-shelf macro, by employing the designer of the macro as a consultant, and by extensive simulation at the board and gate level. The hardware design effort was exemplary; although the "gate count" for the IOU is relatively low, it is rich in functionality (complexity), with interfaces to a processor, an Manchester Encoder-Decoder, and both RAM & ROM. The fact that it uses an 8-bit

Figure 3: IOU Block Diagram

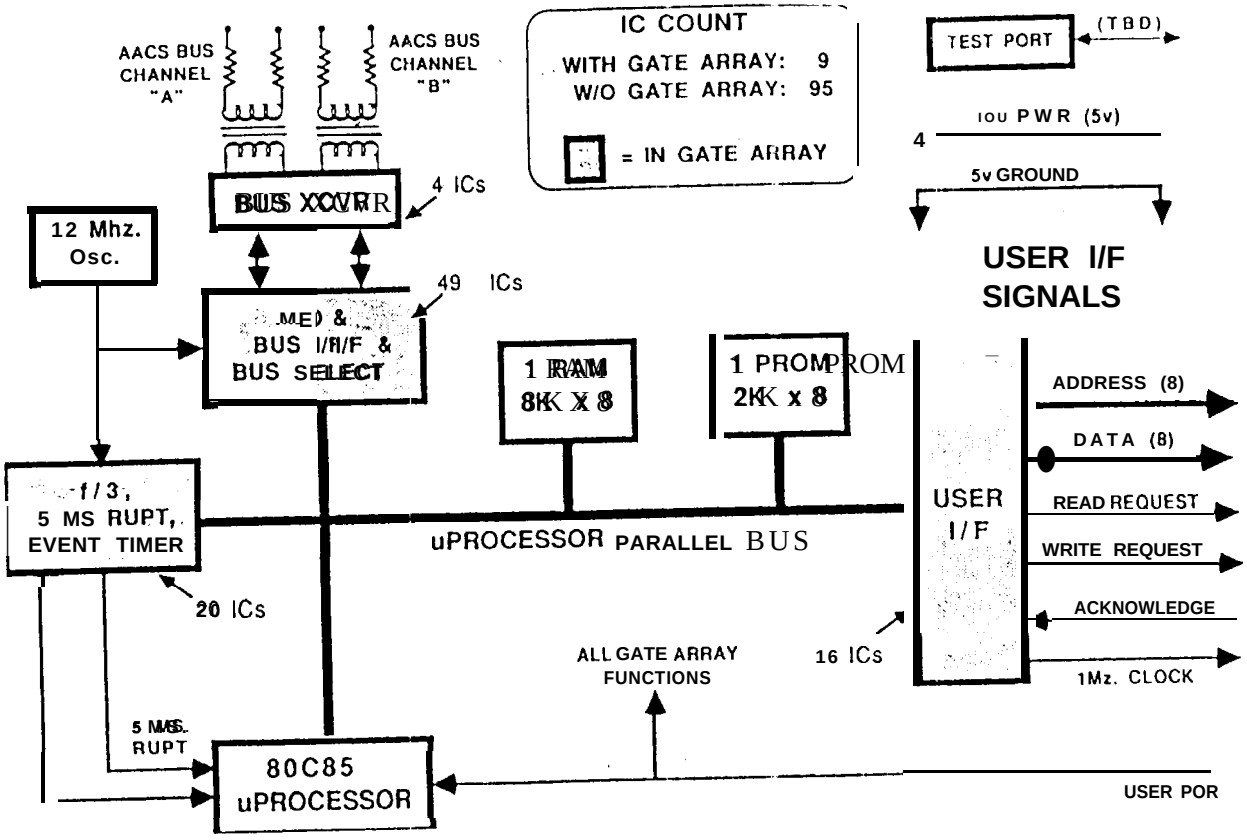
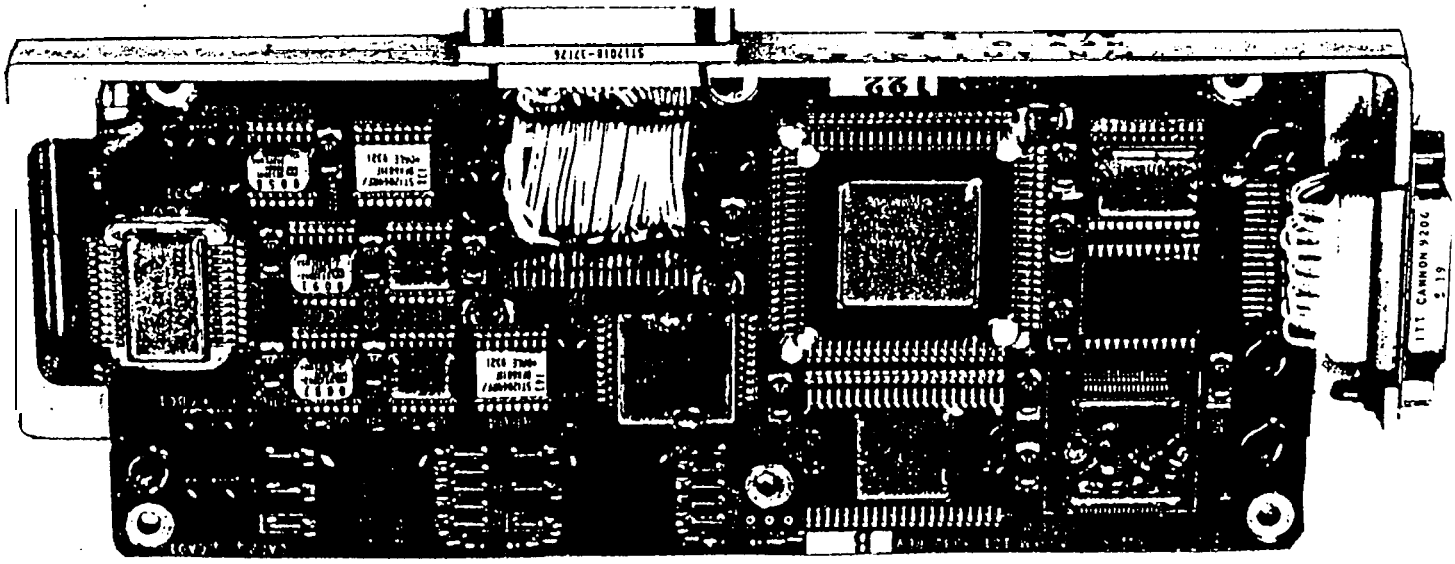


Figure 4: Flight AACS IOU



bus rather than a 16-bit bus also reduces the gate count without reducing the complexity. It was the first gate array produced for the Cassini project, using a new set of CAD tools, and designed by an experienced designer who, however, had no ASIC design experience.

Even though plug-in BC IOU Simulator boards were provided to users, problems were experienced by users in developing their test SW and by being unable to run the PC plug-in board in a particular host machine (due to lack of "universality" in the board design, which was corrected).

The decision to give up bandwidth capability early was **a mistake (one is tempted to say "never** give up bandwidth **!").** The firmware had enough challenges without having to squeeze speed out of an outmoded processor. Too many functions were piled onto the firmware without adequately scoping the task, with the result that not enough planning was done to develop this code, and not enough resources (people, time, and money) were allocated to the effort.

FUTURE DIRECTIONS

The advent of extremely high speed serial buses in the near future, both electrical and optical, will allow the exchange of massive amounts of data and distributed functionality. System nodes will make usc of trends towards more powerful digital and signal processing capability. These future directions validate the approach taken with the IOU.

While it is true in the short term that (in the interests of economy) some spacecraft will knit together off-the-shelf components with custom glue logic, in the long term this cannot be sustained as the "shelf" is depleted. To achieve the ambitious requirements of future missions, high performance components and **I/O will be** essential.

ACKNOWLEDGMENTS

The Cassini AACS IOU is the result of the collaboration of many individuals. The JPL contributors include James Marr IV, who provided the idea in the first place, Jerry Rousey, who was the principal architect, designer, and validator of the unit as it stands today, Edwin Montgomery, who supported hardware **design and test, and also was** solely responsible for the flight firm ware, Gregory Pixler, who implemented the first discrete breadboard version with the assistance of Valerie Stanton, and who also finalized and verified the flight design, Walter Titherington, who designed the printed circuit boards, Gena Lofton, who expedited the procurement of the gate array, Richard Williamson, Technical Manager for the flight implementation phase, Ed Shalom, Cognizant Engineer for AACS Electronics in the development **phase, and many others. Special thanks are also due to employees** from United Technologies Microelectronics Center (UTMC) in Colorado Springs, Colorado for their exemplary role in ASIC design conversion, validation, and fabrication. The technical contributions of Richard Disney **(previously of** UTMC) **and the programmatic support** of Ronald Lake of UTMC merit particular attention.

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.